AF

HEWLETT-PACKARD COMPANY
Intellectual Property Administration
P.O. Box 272400
Fort Collins, Colorado 80527-2400

PATENT APPLICATION

ATTORNEY DOCKET NO. 10002695-1

IN THE
UNITED STATES PATENT AND TRADEMARK OFFICE

Inventor(s): Doug GRUMANN

Application No.: 09/882,845

Filing Date: June 15, 2001

Confirmation No.: 8777

Examiner: Lechi Truong

Group Art Unit: 2194

Title: APPARATUS AND METHOD FOR ENHANCING PERFORMANCE OF A COMPUTER SYSTEM

Mail Stop Appeal Brief-Patents
Commissioner For Patents
PO Box 1450
Alexandria, VA 22313-1450

TRANSMITTAL OF APPEAL BRIEF

Transmitted herewith is the Appeal Brief in this application with respect to the Notice of Appeal filed on 7/10/2007.

The fee for filing this Appeal Brief is (37 CFR 1.17(c)) $500.00.

(complete (a) or (b) as applicable)

The proceedings herein are for a patent application and the provisions of 37 CFR 1.136(a) apply.

☐ (a) Applicant petitions for an extension of time under 37 CFR 1.136 (fees: 37 CFR 1.17(a)-(d)) for the total number of months checked below:

☐ 1st Month $120	☐ 2nd Month $450	☐ 3rd Month $1020	☐ 4th Month $1590

☐ The extension fee has already been filed in this application.

☑ (b) Applicant believes that no extension of time is required. However, this conditional petition is being made to provide for the possibility that applicant has inadvertently overlooked the need for a petition and fee for extension of time.

Please charge to Deposit Account 08-2025 the sum of $500. At any time during the pendency of this application, please charge any fees required or credit any over payment to Deposit Account 08-2025 pursuant to 37 CFR 1.25. Additionally please charge any fees to Deposit Account 08-2025 under 37 CFR 1.16 through 1.21 inclusive, and any other sections in Title 37 of the Code of Federal Regulations that may regulate fees. A duplicate copy of this sheet is enclosed.

☑ I hereby certify that this correspondence is being deposited with the United States Postal Service as first class mail in an envelope addressed to: Commissioner for Patents, Alexandria, VA 22313-1450

Date of Deposit: 09/10/2007

OR

☐ I hereby certify that this paper is being transmitted to the Patent and Trademark Office facsimile number (571)273-8300.

Date of facsimile:

Typed Name: Ilene L. Fish

Signature: Ilene L. Fish

Respectfully submitted,
Doug GRUMANN

By

John P. Wagner, Jr.

Attorney/Agent for Applicant(s)

Reg No. : 35,398

Date : 09/10/2007

Telephone : 408-377-0500

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

Appellant:	Grumann et al.	Patent Application	
Serial No.:	09/882,845	Group Art Unit:	2194
Filed:	June 15, 2001	Examiner:	Truong, Lechi

For: APPARATUS AND METHOD FOR ENHANCING PERFORMANCE OF A COMPUTER SYSTEM

Appeal Brief

09/13/2007 HVUONG1 00000053 082025 09882845
01 FC:1402 500.00 DA

Table of Contents

Real Party in Interest

The assignee of the present invention is Hewlett-Packard Company.

Related Appeals and Interferences

There are no related appeals or interferences known to the Appellant.

Status of Claims

Claims 1-26 remain pending. Claims 1-26 have been rejected. This appeal involves Claims 1-26.

Status of Amendments

All proposed amendments have been entered. An amendment subsequent to the Final Action has not been filed.

Summary of Claimed Subject Matter

Independent Claims 1, 12, 14 and 24 pertain to various embodiments for enhancing performance of a computer system. For example, independent Claims 1 and 12 recite,

> electronically deriving relationships over time between monitored system variables and monitored performance of said computer system;
> automatically generating a number of rules based on said derived relationships, wherein said number of rules are generated without requiring human interaction; and
> adjusting at least one of said system variables based on said generated number of rules to enhance the performance of said computer system.

Independent Claim 14 recites,

> program code for deriving relationships over time between monitored system variables and monitored performance of said computer system;
> program code for automatically generating a number of rules based on said derived relationships, wherein said number of rules are generated without requiring human interaction; and
> program code for adjusting at least one of said system variables based on said generated number of rules to enhance the performance of said computer system.

Independent Claim 24 recites,

> means for electronically deriving relationships over time between monitored system variables and monitored performance of said computer system;
> means for automatically generating a number of rules based on said derived relationships, wherein said number of rules are generated without requiring human interaction; and
> means for adjusting at least one of said system variables based on said generated number of rules to enhance the performance of said computer system.

Various embodiments recited by the independent Claims 1, 12, 14 and 24 are described at page 5 line 28 to page 6 line 2 of the instant application serial no. 09/882,845, which states with reference to Figures 1 and 5,

> The system performance 120 may be determined based on a number of performance metrics 121-123. The invention enhances the performance of the computer system 100, generally, by deriving relationships between the system variables 110 and the system performance 120, generating a number of (ie., one or more) rules 500-502 (FIG. 5) based thereon, and adjusting at least one of the system variables 110 based on one or more of the rules 500-502 (emphasis added).

The following is a description of "deriving relationships over time between monitored system variables and monitored performance of said computer system," according to one embodiment, as recited by the independent Claims. Page 6 lines 4-7 provide examples of system variables 110 such as the number of active CPUs, the amount of physical memory allocated to disk caching, variables that may affect the performance of the computer system 100, and a particular database schema indexing policy configuration parameters. Table 1 at the bottom of page 10 provides other examples of system variables. Page 7 lines 7-8 indicate that system performance may include performance metrics 121-123. Page 6 lines 11-13 state, "Exemplary performance metrics 120 may be measured in terms of service health, application response time, throughput, etc." Table 2 on page 11 lines 10-13 depicts other examples of performance metrics.

According to one embodiment, "...the data for the system variables 110 and the system performance 120 may be acquired by any suitable data acquisition..." as stated on page 7 lines 32-33. An example of gathered data is depicted at page 8 line 25 to page 9 line 13 as follows:

System Variables (110) at discrete point in time T1

Application parameter XYZ = 35
Application Buffer Size = 2345 kB
Application Instance Count = 3
Application Parameter JKL = 104
...
Active Processor Count = 4
Physical Memory Size = 2 GB
Network Daemon Count = 8
Disk Raid Value = 5
Buffer Cache Maximum = 1024 MB
...

System Performance (120) at discrete point in time T1

App1 Response Time = 1.3 sec
App1 Throughput = 12 per sec
App 2 Response Time = 0.05 sec

...

Average Service Time = 0.13 sec
User Count = 207
...

The instant application states at page 11 lines 24-26 with reference to Figures 1 and 4, "Exemplary data acquired for each of the system variables 110 and performance metrics 120 is provided in Table 3 at seven separate times, T1-T7." Table 3 is depicted at the top of page 12 of the instant application.

Referring to page 9 line 32-page 10 line 1, the instant application states that Figure 4 of the instant application depicts "a series of plots 400, 401 and 450, 451 of historical data 420, 421, and 470, 471 illustrating the relationships between the system variables 110 and the system performance 120..." Referring to Tables 1-3 in the instant application and Figure 4, the instant application states at page 13 lines 12-23,

> An exemplary analysis, wherein the relationships between the system variables 110 and the system performance 120 are derived, may be as follows, based on the acquired data shown in the above Tables 1-3. When SysVar1 is increased and SysVar2 is held constant at 1000 MB, both PerfMetric1 and PerfMetric2 improve (i.e., tend toward the respective performance goals). When SysVar2 is then increased while SysVar1 is held constant, PerfMetric1 deteriorates (i.e., tends away from the performance goal) while PerfMetric2 continues to improve. The relationships derived from this data would hold PerfMetric1 as strongly inversely proportional to values SysVar1 and less strongly proportional to values of SysVar2. PerfMetric is inversely proportional to both SysVar1 and SysVar2.

The following is a description of "automatically generating a number of rules based on said derived relationships, wherein said number of rules are generated without requiring human interaction," according to one embodiment, as recited by the independent Claims. At page 14 line 17 to page 15 line 8, the instant application states with reference to Figure 4 and Tables 1-3,

> As an example, based on the acquired data in Tables 1-3, the first system variable 110 has a range from 10 MB to 500 MB, and the second system variable 110 has a range from 1000 MB to 10000 MB. Therefore, any rules 500 must be within the respective ranges. The analysis of the historical data indicates that when SysVar1 is 500 MB, both PerfMetric1 and PerfMetric2 tend toward the respective performance goals when

SysVar 2 is between 200) MB and 4000 MB. Therefore, exemplary rules 500-502, may comprise:

IF SysVar <> 500 MB,
    Then SetSysVar1 = 500 MB.
IF SysVar2 < 2000 MB or SysVar 2 > 4000 MB),
    Then Set SysVar2 = 3000MB.

Other exemplar rules based on the analysis of the data in Tables 1-3 may include the following:

If PerfMetric1 > Goal1 (1),
    Then Increase SysVar1 by Increment (50) limit Max1 (500)
    And Decrease SysVar2 by Increment (100) limit Min2 (1000).
If PerfMetric2 > Goal2 (4),
    Then Increase SysVar1 by Increment (50) limit Max1 (500)
    AND Increase SysVar2 by Increment (100) limit Max2 (10000).

The following is a description of "adjusting at least one of said system variables based on said generated number of rules to enhance the performance of said computer system," according to one embodiment, as recited by the independent Claims. At page 16 lines 8-10, the instant application states with reference to Figures 1 and 5, "The rules 500-502 may specify at least one adjustment 520-522 to be made to the computer system 100 when a condition 510-512 is met or approached."

Grounds of Rejection to be Reviewed on Appeal

1. In paragraph 4 of the Office Action, Claims 1-3, 5-15 and 17-26 are rejected under 35 U.S.C. §103(a) as being unpatentable over U.S. patent no. 6,059,842 by Dumarot et al. (referred to herein as "Dumarot") in view of U.S. patent no. 6,144,954 by Li (referred to hereinafter as "Li").

2. In paragraph 25 of the Office Action, Claims 4 and 16 are rejected under 35 U.S.C. § 103(a) as being unpatentable over Dumarot in view of Li and further in view of Mihata (JP403010379).

Arguments

1. Whether Claims 1-3, 5-15, and 17-26 are unpatentable under 35 U.S.C. 103(a) over Dumarot (6,059,842) in view of Li (6,144,954)

A. Scope and Content of the Cited Prior Art Reference (Dumarot and Li)

Dumarot teaches a way of optimizing. Figure 5 provides a description of various embodiments taught by Dumarot for optimizing software applications. At Col. 5 line 29 to Col. 6 line 37, Dumarot states,

> At step 303, the optimizer 136 gathers relevant system information.... At step 305, the optimizer 136 gathers relevant application information. ... At step 310, the optimizer 136 reads records 530 from database 140, that control various parameters 420, associated with a particular application name 410... In step 320, the optimizer 136 monitors system 12 behavior...This activity 321 may be stored in the form of dynamic values M1, M2...M2 in settings 460 and read by the optimizer program 136...In step 325, the optimizer 136 reads user input. For example, the user may enter text or data at the keyboard 40...that specifies a level of optimization...This level of optimization may control which of the application settings 420 are used to optimize the application in step 330 or optimize the system 12 in step 340. A user wishing to have maximum performance may, for example, sacrifice graphic quality controlled in applications settings 420, that are generally read upon invocation of application 138. ...In this example, if a user sets suppressAutoRefresh to TRUE, the application performance can improve by reducing excess redrawing (emphasis added).

Referring to Col. 6 lines 46-47, various embodiments of Dumarot can also be used to optimize system settings. For example, at Col. 6 line 68 to Col. 7 line 56 Dumarot states,

> Generally speaking, in steps 330 and 340, the optimizer uses the information acquired in steps 303, 305, 310, 320, and 325 to adjust system or application parameters in order to optimize the operation of the application. ... In general, the optimizer adjusts system and application settings to best meet user-specified quality/performance trade-offs....The database can be helpful in determining changes to system and application configurations at different points in time, in evaluating the effects of changing application settings, and in comparing actual system/application settings with recommended settings. ... The recommendations may be used to warn the user of various conditions (e.g., "disk space is low"), or give suggestions on how to improve performance (e.g., "purchase more memory"). The optimizer contains rules 331, 341, 351 that it uses to make such optimizations 330, 340 and recommendations 350...Additionally, the rules 331, 341, 351 may require password

protection so that only certain users or classes of users have permission to implement the rules (emphasis added).

Li provides various embodiments to address the performance maintenance tradeoff associated with embedded knowledge bases. Li states in the abstract,

> A self-optimizing method and machine automatically develop computer software in real-time according a specified performance by computer-generating a knowledge base associated with the computer software, by instantly computer-coding the computer-generated knowledge base into the computer software, and by saving the developed computer software in a software storage device (emphasis added).

B. Differences Between the Cited Prior Art References and the Claimed Invention.

Independent Claim 1 recites,

> A computer-implemented method for enhancing performance of a computer system, comprising:
> electronically deriving relationships over time between monitored system variables and monitored performance of said computer system;
> automatically generating a number of rules based on said derived relationships, wherein said number of rules are generated without requiring human interaction; and
> adjusting at least one of said system variables based on said generated number of rules to enhance the performance of said computer system.

Appellants respectfully assert that Dumarot does not teach or suggest, "electronically deriving relationships over time between monitored system variables and monitored performance of said computer system; automatically generating a number of rules based on said derived relationships, wherein said number of rules are generated without requiring human interaction," as recited by Claim 1.

As indicated by the underlined portions of Col. 5 line 29 to Col. 6 line 37 quoted in section 1.A above, note that Dumarot teaches that if the user wishes to maximize performance, the user will specify the setting for an application parameter that will result in maximizing performance.

With reference to Col. 6 line 68 to Col. 7 line 56 quoted in section 1.A above, note that Dumarot's quality/performance trade offs are user specified. The user specified quality/performance trade offs drive the types of things the system will recommend to the user. Also note that Dumarot requires human interaction to generate his rules since Dumarot's rules are only implemented by a user or certain classes of users.

The Office Action asserts that Dumarot teaches "electronically deriving relationships over time between monitored system variables and monitored performance of said computer system," as recited by Claim 1 at 330, 341, 351, Col. 7 lines 25-35, Col. 7 lines 5-16, Col. 7 lines 10-16, and Col. 5 lines 10-17. The Office Action states in the first sentence of the fifth paragraph of the Office Action, "...Dumarot teaches the invention substantially as claimed including: electronically deriving relationships (the optimizer contains rules 330, 341, 351...). Further, all of these cited portions of Dumarot refer to "rules." Therefore, it appears that in paragraph 5, the Office Action is asserting that Dumarot's rules teach Claim 1's "relationships." However, even assuming for the sake of argument that Dumarot's "rules" are analogous to Claim 1's "relationships" (this is not an admission on the part of Appellants) note that Dumarot's "rules" are only user specified as indicated, among other places, by Col. 7 lines 54-56 of Dumarot. Then a user can select a rule as indicated in many portions of Dumarot such as the abstract, Col. 3 lines 21-22, Col. 3 lines 43-46, Col. 4 lines 49-50, Col. 7 lines 39-43, and Co. 8 lines 26-33.

Therefore, even assuming for the sake of argument that Dumarot's "rules" are analogous to Claim 1's "relationships," Dumarot still does not teach or suggest "electronically deriving relationships over time between monitored system variables and monitored performance of said computer system." Further, by teaching a system that utilizes and requires a user to implement rules, Dumarot teaches away from "electronically deriving relationships over time," (emphasis added).

The Office Action asserts that Dumarot teaches "electronically deriving relationships over time," (emphasis added) at Col. 7 lines 10-16 and Col. 5 lines

10-17. However, Dumarot teaches at Col. 7 lines 10-16 "determining changes to system and application configurations at different points in time." This is not "electronically deriving relationships over time," (emphasis added). Col. 5 lines 10-17 refer to "dynamic data." It appears to Appellants that the Office Action is asserting that Dumarot's "dynamic data" is analogous to Claim 1's "relationships" which are electronically derived over time. However, Dumarot states in Col. 4 lines 7-10, "The dynamic data is generally dynamic information, such as current CPU, memory, and disk use, all of which change as an application performs operations, and reads and writes information to memory and disk." Therefore, Col. 5 lines 7-10 make it clear that Dumarot's "dynamic data" does not teach or suggest anything about "relationships" let alone teach or suggest "electronically deriving relationships over time."

Since Dumarot does not teach "electronically deriving relationships over time between monitored system variables and monitored performance of said computer system," Dumarot cannot teach "automatically generating a number of rules based on said derived relationships, wherein said number of rules are generated without requiring human interaction." Further, note that Dumarot teaches directly away from "automatically generating a number of rules based on said derived relationships, wherein said number of rules are generated without requiring human interaction" because Dumarot requires human interaction to determine Dumarot's rules. For example, on Col. 7 lines 54-56, Dumarot states, "...the rules 331, 341, 351 may require password protection so that only certain users or classes of users have permission to implement the rules" (emphasis added).

In the **Response to Argument's Section**, the Office Action states, "comparing [relationships] actual system/application settings with recommended settings [monitored system variables and monitored performance] (col. 7, ln 11-16)..." It appears that the Office Action is asserting that Dumarot's "comparing actual system/application settings with recommended settings teach Claim 1's "relationships." Dumarot states at Col. 7 lines 9-19,

> In general, the optimizer adjusts system and application settings to best meet user-specified quality/performance trade-offs. The information gathered in steps 303, 305, and 320 may be stored in the database 140 maintained by the optimizer. The database can be helpful in determining

changes to system and application configurations at different points in time, in evaluating the effects of changing application settings, and in comparing actual system/application settings with recommended settings. (emphasis added).

In step 350, the optimizer 136 may provide suggestions or recommendations 480, for example, in the form of specific text that is output to the user.

Claim 1 recites, "relationships ...between monitored system variables and monitored performance of said computer system..." Dumarot's "recommended settings" referred to at Col. 7 line 16 does not teach or suggest either of Claim 1's "monitored system variables" or Claim 1's "monitored performance." Therefore, Dumarot cannot teach or suggest, "electronically deriving relationships over time between monitored system variables and monitored performance of said computer system," as recited by Claim 1.

Dumarot goes on to state at Col. 7 lines 55-56, "...only certain users or classes of users have permission to implement the rules." Further, at Col. 8 lines 13-18, Dumarot states, "Note that one benefit of having portions of the database 140 (the settings and suggestions) and rules 331, 341, and 351 on a remote machine 130 is that a company or system administrator can continually manage and update messages and rules as new information is provided by application vendors." Therefore, Dumarot cannot teach or suggest, "automatically generating a number of rules based on said derived relationships..." In fact, by teaching that "only certain users or classes of users have permission to implement the rules, "Dumarot teaches directly away from "automatically generating a number of rules based on said derived relationships...," as recited by Claim 1.

Li cannot be used to remedy the deficiency in Dumarot in that neither Li nor Dumarot teach or suggest "electronically deriving relationships over time between monitored system variables and monitored performance of said computer system; automatically generating a number of rules based on said derived relationships, wherein said number of rules are generated without requiring human interaction," as recited by Claim 1.

Li states in the background section at Col. 1 lines 13-19,

> Computer-controlled automation systems are widely used. These systems are extremely powerful. With their uses, consistency is achieved together with the usually, but not necessarily always, associated improved profits, productivity, and product or service qualities (P^3Q).
>
> Further, by transferring to machines human intelligence, rather than skill, these systems have ushered us into a Second Industrial Revolution.

Li also states in the background section at Col. 1 lines 56-63,

> Without reliable knowledge bases, the usual automation specialists would be at a loss in selecting the few among many manufacturing or servicing phenomena or variables to be controlled, as well as in formulating the system dynamics models, in setting up the control equations, in determining the control constraints, and in specifying setpoints for the control variables.

Lastly Li states in the background section at Col. 3 lines 14-22,

> Conventional AI development environments experience difficulties in producing efficient real-time systems. This is due to the fact that the same code necessary to enhance the development environment tends to slow down the system during run-time. To overcome these limitations, AI system designers must embed a knowledge base (KB) into their own custom run-time AI shells to achieve real-time performance. Unfortunately, the deeper the KB is embedded into the actual code for improved performance, the harder it is to change the KB when maintenance is necessary.

Li provides various embodiments to address the performance maintenance tradeoff associated with embedded knowledge bases. Li states in the abstract,

> A self-optimizing method and machine automatically develop computer software in real-time according a specified performance by computer-generating a knowledge base associated with the computer software, by instantly computer-coding the computer-generated knowledge base into the computer software, and by saving the developed computer software in a software storage device (emphasis added).

The Office Action does not assert that Li teaches "electronically deriving relationships over time between monitored system variables and monitored performance of said computer system." Further, Li cannot be combined with Dumarot to teach "electronically deriving relationships over time between monitored system variables and monitored performance of said computer system" because both Dumarot and Li teach away from "electronically deriving relationships over time between monitored system variables and monitored

performance of said computer system." For example at Col. 4 lines 53-58 Li states,

> setting the m variables to the thus-computed optimal combinations before these combinations change; and feeding back information on the status of optimization to achieve closed-loop feed-back control. The knowledge bases so generated are instantly machine-coded. (emphasis added)

Since Li teaches feeding the "combinations" before they change and instantly machine coding based on those combinations, Li teaches away from "electronically deriving relationships over time between monitored system variables and monitored performance of said computer system," as recited by Claim 1.

Col. 19 lines 58-65 is another example of Li teaching away from "electronically deriving relationships over time between monitored system variables and monitored performance of said computer system." At Col. 19 lines 58-65, Li states,

> Again for the self-optimizing machine, the software optimizing task such as for software generation, usage or maintenance, computer simulation, and computer aided design (CAD), engineering (CAE), or testing (CAT) etc. is first defined. The optimizing criteria of the usual cost, productivity, and quality, in different weighing indices are next given. The number, type, and allowable range of the variables in different categories are then fed. (emphasis added)

Since Li does not teach or suggest "electronically deriving relationships over time between monitored system variables and monitored performance of said computer system" Li cannot teach or suggest "automatically generating a number of rules based on said derived relationships, wherein said number of rules are generated without requiring human interaction."

Li cannot be combined with Dumarot because both teach away from the embodiment recited by Claim 1. In order to establish a *prima facie* case of obviousness, the prior art must suggest the desirability of the claimed invention (MPEP 2142). In particular, "[i]t is improper to combine references where the references teach away from their combination" (emphasis added; MPEP 2145(X)(D)(2); In re Grasseli, 713 F.2d 731, 743, 218 USPQ 679, 779 (Fed. Cir. 1983)). Appellants respectfully note that "[a] prior art reference must be considered in its entirety, i.e., as a whole, including portions that would lead

away from the claimed invention" (emphasis in original; MPEP 2141.02(VI); W.L. Gore & Associates, Inc. v. Garlock, Inc., 721 F.2d 1540, 220 USPQ 303 (Fed. Cir. 1983), cert. denied, 469 U.S. 851 (1984)). Appellants submit that there is no motivation to combine the teachings of Dumarot and Li because both Dumarot and Li teach away from "electronically deriving relationships over time between monitored system variables and monitored performance of said computer system," as recited by Claim 1.

Further, Li cannot be combined with Dumarot because the teachings of Li would render the teachings of Dumarot unsatisfactory for their intended purpose. MPEP 2143.01 states, "if the proposed modification or combination of the prior art would change the principle of operation of the prior art invention being modified, then the teachings of the references are not sufficient to render the claims *prima facie* obvious" (emphasis added) (MPEP 2143.01; in re Ratti, 270 F.2d 810, 123 USPQ 349 (CCPA 1959)). Moreover, "[i]f the proposed modification would render the prior art invention being modified unsatisfactory for its intended purpose, then there is no suggestion or motivation to make the proposed amendment" (emphasis added) (MPEP 2143.01; in re Gordon, 733 F2.d 900, 221 USPQ 1125 (Fed. Cir. 1984)). For example, since Li teaches "The knowledge bases so generated are instantly machine-coded" the teachings of Li would render teachings of Dumarot, such as the Dumarot's teachings that enable users to implement rules, unsatisfactory for Dumarot's intended purpose.

Therefore, Claim 1 should be patentable over Dumarot and Li, alone or in combination, for at least the reasons that neither Dumarot nor Li teach or suggest," electronically deriving relationships over time between monitored system variables and monitored performance of said computer system; automatically generating a number of rules based on said derived relationships, wherein said number of rules are generated without requiring human interaction." Independent Claims 12, 14, and 24 should be patentable for similar reasons that Claim 1 should be patentable. Claims 2-11 depend on Claim 1. Claim 13 depends on Claim 12. Claims 15-23 depend on Claim 14. Claims 25-26 depend on Claim 24. These dependent claims include all of the features of their respective independent claims. Further these dependent claims include additional features which further make them patentable. Therefore, these

dependent claims should be patentable for at least the reasons that their respective independent claims should be patentable.

In summary, Appellants respectfully submit that the Office Action's rejections of the claims are improper as the rejection of Claims 1-3, 5-15, and 17-26 does not satisfy the requirements of a prima facie case of obviousness as claim features are not met by the cited references and there is no motivation to combine the references. Accordingly, Appellants respectfully submit that the rejection of Claims 1-3, 5-15, and 17-26 under 35 U.S.C. §103(a) are improper and should be reversed.

2. Whether Claims 4 and 16 are unpatentable under 35 U.S.C. 103(a) over Dumarot (6,059,842) in view of Li (5,144,954) further in view of Mihata (JP403010379A).

A. Scope and Content of the Cited Prior Art References (Dumarot, Li, and Mihata)

The scope of Dumarot has already been discussed in section 1.A

The scope of Li has already been discussed in section 1.A.

Referring to the title, Mihata teaches a design rule verifying system. For example, referring to the abstract, Miahata states,

> PURPOSE: To improve the efficiency of a correcting work by the design rules to a free bit in a contradictory rule storage rea, in case plural pieces of contradictory design rules are generated, and enumerating and displaying them in a batch.
>
> CONSTITUTION: In the case plural pieces of contradictory design rules are generated, those design rules are allocated to a free bit in order in a contradictory rule storage area 119 by a storage area allocating means 125, and also, the contradictory design rules are enumerated and displayed in a batch at every verified design data on a display screen of a graphic processor by a contradictory rule display means 126. In such a way, even in case plural pieces of contradictory design rules exist, such an inconvenience as a verification processing by a computer and a correcting work to the contradictory design rule are repeated extending over a large number of times due to being unfamiliar with the contradictory contents can be evaded, the efficiency of the correcting work can be improved.

B. Differences Between the Cited Prior Art References and the Claimed Invention.

Mihata is a Japanese Patent. MPEP 706.02 II states that it is the Examiner's responsibility to provide a translation to references that the Examiner is relying on. Although the Appellants have requested a translation of Mihata in every response that the Appellants have filed, the Examiner has failed to provide the translation. The only part of Mihata that has been provided in English is the abstract. Based on the abstract, it appears that Mihata does not remedy the deficiency in Dumarot and Li. For example, Claim 4 recites, "said generating said number of rules, and said adjusting said at least one system variable, are iterative." In contrast Mihata recites "contradictory design rules." Further, the

Office Action has not cited Mihata against the independent Claims 1 and 15 that Claims 4 and 16 depend on respectively. Therefore, Appellants respectfully submit that Claims 4 and 16 are patentable over Dumarot, Li and Mihata, alone or in combination.

In summary, the Appellant respectfully requests that the Board reverse the Examiner's rejections of Claims 1-26.

The Appellant wishes to encourage the Examiner or a member of the Board of Patent Appeals to telephone the Appellant's undersigned representative if it is felt that a telephone conference could expedite prosecution.

Respectfully submitted,

WAGNER BLECHER LLP

Date: 9/10/2007

John P. Wagner

Registration Number: 35,398

WAGNER BLECHER LLP
Westridge Business Park
123 Westridge Drive
Watsonville, CA 95076
408-377-0500

Claims Appendix

1. (Previously Presented) A computer-implemented method for enhancing performance of a computer system, comprising:

electronically deriving relationships over time between monitored system variables and monitored performance of said computer system;

automatically generating a number of rules based on said derived relationships, wherein said number of rules are generated without requiring human interaction; and

adjusting at least one of said system variables based on said generated number of rules to enhance the performance of said computer system.

2. (Previously Presented) A computer-implemented method as in claim 1, wherein said generating said number of rules is based at least in part on a performance goal.

3. (Previously Presented) A computer-implemented method as in claim 1, wherein said generating said number of rules is based at least in part on current values of said system variables, and wherein said number of rules recommend incremental changes to said system variables.

4. (Previously Presented) A computer-implemented method as in claim 1, wherein said deriving said relationships, said generating said number of rules, and said adjusting said at least one system variable, are iterative.

5. (Previously Presented) A computer-implemented method as in claim 1, further comprising acquiring data for said system variables and the performance of said computer system, wherein said acquired data is used for deriving said relationships.

6. (Previously Presented) A computer-implemented method as in claim 5, wherein acquiring said data comprises:

gathering said data over time; and

logging said gathered data, wherein said relationships are derived based on said logged data.

7. (Previously Presented) A computer-implemented method as in claim 6, wherein said gathering said data is at discrete points in time.

8. (Previously Presented) A computer-implemented method as in claim 6, wherein said gathering said data is in response to an event on said computer system.

9. (Previously Presented) A computer-implemented method as in claim 5, wherein said acquiring said data comprises acquiring at least one of the following types of data: configuration data, workload data, and performance metric data.

10. (Previously Presented) A computer-implemented method as in claim 1, further comprising identifying a number of applications on said computer system having variables that affect the performance of said computer system.

11. (Previously Presented) A computer-implemented method as in claim 1, further comprising identifying a number of subsystem components on said computer system having variables that affect the performance of said computer system.

12. (Previously Presented) A computer-implemented method for enhancing performance of a computer system, comprising:

electronically deriving a plurality of relationships over time between a plurality of monitored system variables and monitored performance of said computer system;

automatically generating a plurality of rules based on said plurality of derived relationships, wherein said plurality of rules are generated without requiring human interaction; and

adjusting at least one of said system variables based on said generated plurality of rules to enhance the performance of said computer system.

13. (Previously Presented) A computer-implemented method as in claim 12, wherein the performance of said computer system is based on a plurality of performance metrics.

14. (Previously Presented) An apparatus for enhancing performance of a computer system, comprising:

computer readable storage media;

computer readable program code stored on said computer readable storage media, comprising:

program code for deriving relationships between system variables and the performance of said computer system;

program code for automatically generating a number of rules based on said derived relationships, wherein said number of rules are generated without requiring human interaction; and

program code for adjusting at least one of said system variables based on said generated number of rules to enhance the performance of said computer system.

15. (Original) An apparatus as in claim 14, wherein said number of rules are generated by said program code based at least in part on a performance goal.

16. (Original) An apparatus as in claim 14, further comprising program code for iteratively deriving relationships between said system variables and the performance of said computer system, and iteratively generating a number of rules based on said derived relationships when an adjustment is made to said at least one system variable.

17. (Original) An apparatus as in claim 14, further comprising program code for acquiring data for said system variables and the performance of said computer system.

18. (Original) An apparatus as in claim 17, wherein at least some of said data is acquired from a configuration file.

19. (Original) An apparatus as in claim 17, wherein at least some of said data is acquired by monitoring said computer system.

20. (Previously Presented) An apparatus as in claim 17, wherein said program code for acquiring said data comprises:

program code for gathering said data over time;

program code for logging said gathered data, wherein said program code for deriving derives said relationships based on said logged data.

21. (Original) An apparatus as in claim 17, wherein said program code for acquiring said data acquires at least one of the following types of data: configuration data, workload data, and performance metric data.

22. (Original) An apparatus as in claim 14, further comprising program code for identifying a number of applications on said computer system having variables that affect the performance of said computer system.

23. (Original) An apparatus as in claim 14, further comprising program code for identifying a number of subsystem components of said computer system having variables that affect the performance of said computer system.

24. (Previously Presented) An apparatus for enhancing performance of a computer system, comprising:

means for electronically deriving relationships over time between monitored system variables and monitored performance of said computer system;

means for automatically generating a number of rules based on said derived relationships, wherein said generated number of rules are generated without requiring human interaction; and

means for adjusting at least one of said system variables based on said generated number of rules to enhance the performance of said computer system.

25. (Original) An apparatus as in claim 24, further comprising means for acquiring data for said system variables and the performance of said system.

26. (Previously Presented) An apparatus as in claim 25, wherein said acquiring means comprises:

means for gathering said data over time; and

means for logging said data, wherein said relationships are derived based on said logged data.

Evidence Appendix

None

Related Proceedings Appendix

None